Lab01

簡述：

用verilog寫出HASH演算法中的message schedule階段的design並且用tb驗證，跑lint, sim, syn，最後比面積當ranking。

心得：

這是我第一次寫verilog，但寫出design本身並不難，而需要考慮面積大小，因此需要簡化用電路實現該演算法的邏輯，這是第一個遇到的困難，在還未routing前的階段想要減少面積主要靠的是減少resource的使用，我在其中認為最主要的是大位元寬的register、多input並重複出現的mux以及大位元寬的加法器，後兩者常常是trade off關係，想要減少加法器就需要在其input處加上選擇邏輯(mux)，這可能需要靠都實作看看或是經驗法則判斷，另外多蚵串連的加法器就不必簡化成單顆，理論上synthesizer是可以優化這種多input的加法的。在縮小面積的路上還會遇到的困難是，明明寫的邏輯是一樣的，換個寫法寫(if nest/case/?:)，面積卻差很多，或甚至是有時候拿掉或加上無意義的default，面積也差很多，助教也不明白為甚麼。

優化面積舉例，64個32-bit reg簡化成16個，應用其遞迴的關係配合上shift reg；合用使用實踐不重疊的counter。

第二個是符合protocol比想像中難，而切初次碰這個語言，花了一些時間熟悉waveform和setup/hold time的關係。

第三個遇到的困難是寫testbench，第一次接觸這個design flow的我花了很多時間才搞懂數位電路設計的flow，這次採用的是BFM的方式寫，其理念在於先用高階語言產生test pattern和golden vector並讀進testbench中的tx和checker，並由tx和rx控制I/O protocol，tx喂input，rx接output，checker檢查，BFM的精隨我認為在於tx要寫得好，讓protocol的各種可能都發生，使得各個corner都跑得到(當然也要test pattern配合)，另一好處在於可以用test case控制。

Lab02

簡述：

用verilog寫出HASH演算法的design並且用tb驗證，跑lint, sim, syn，最後比面積當ranking。

心得：

這次的design大很多，且I/O protocol複雜很多，其實在上一個lab還有上課過程中我就已經體認到planning的重要性，所以其實時做出這個電路也沒有遇到太大的困難，在debug部分運用verdi追蹤driver和nwave波型圖可以糾錯蠻快的(可能沒有甚麼bug，所以也de的蠻有成就感的，因為很快就可以work)，這次我認為最大的困難是testbench的部分，transmitter要做的事情太多了，要設定各種情況以驗證design有沒有遵守protocol，還有非常多input訊號要給，因此我是在裡面設計了一個FSM來控制。在這個lab還有學到用wavedrom畫waveform的技能。

Lab03

簡述：

將lab02寫出的電路燒進FPGA板子，實際用硬體模擬驗證此電路之功能，使用vivado的系統完成此次的demo，demo內容是在電腦直接輸入一串英文(ASCII)，設計要能夠送回正確之hash value。

心得：

這個過程大部分是照著TA給的指示去操作，需要自行設計的部分包含第一個：design與AXI介面間的管理與交流，以及第二個：使用C語言完成使用者介面與硬體間的交流，也就是flow control以及data的傳送接收。第一個部份我的solution是再設計一個block(包含FSM在其中控制)夾在兩者之間當作interface去做訊號轉換與控制，在demo時聽完其他人的做法後，我認為我的方法是很容易理解並且執行的，因為如果不在硬體端處理好訊號的轉換與控制的話，軟體端會變得太過複雜，最終心得是兩邊都需要各司其職完成該做的是達到一個平衡才能快速簡單的完成這項任務。第二個部份我遇到的困難是我第一次寫C code，所以有很多剛接觸的知識，還有涉及到兩個CPU之間的交流時，需要考慮時脈等等的不同。

學期心得：

從入門verilog的語法，培養好的coding style，到完成一個演算法的電路實現並驗證，在這過程中學習到非常多design與verification的基礎知識，實作過程也非常充實，包含用rtl實現演算法以及建立完整的testbench(BFM)去驗證功能性。